CRIPTANALIZA. REZULTATE ŞI TEHNICI MATEMATICE

Emil SIMION

Cuprins

1 CRIPTANALIZA CIFRURILOR BLOC **5**

1.1. Introducere şi concepte generale 5
1.2. Securitatea şi complexitatea atacurilor 6
1.3. Criterii de evaluare a cifrurilor bloc 6
1.4. Moduri de operare 6
1.4.1. Modul ECB (electronic code-book) 7
1.4.2. Modul CBC (cipher-block chaining) 8
1.4.3. Modul CFB (cipher feedback) 11
1.4.4. Modul OFB (output feedback) 12
1.4.5. Modul BC (block chaining) 14
1.4.6. Modul BC cu sumă de control (BC-checksum) 14
1.4.7. Modul OFBNLF (output feedback block with a nonlinear function) 14
1.4.8. Cascade de cifruri şi cifrări multiple 15
1.5. Generarea tabelelor de substituţie 18
1.6. Criptanaliza diferenţială 19
1.7. Criptanaliza liniară 19
1.8. Alte metode 19
1.9. Implementări şi rezultate experimentale 20
1.9.1. Implementarea standardului de cifrare A.E.S. 20
1.9.2. Testarea algoritmului AES 21
1.9.3. Rezultate experimentale 21
1.9.4. Interpretarea rezultatelor 24
1.10. Concluzii 24
1.11. Aplicaţii 25

Capitolul 1

CRIPTANALIZA CIFRURILOR BLOC

Nearly every inventor of a cipher system was been convinced of the unsolvability of his brain child.
David Kahn

1.1. Introducere şi concepte generale

Prezentul capitol face o prezentare a noţiunii de cifru bloc, a modurilor de operare precum şi a principalelor caracteristici ale acestora. În finalul capitolului se prezintă o serie de tehnici şi metode de criptanaliză a cifrurilor bloc care vor fi exemplificate pe noul standard de cifrare bloc AES (*Advanced Encryption Standard*).

Cifrurile bloc procesează informaţia pe blocuri de o lungime stabilită apriori. În cele ce urmează vom nota prin n lungimea blocului procesat (exprimată în biţi), V_n spaţiul vectorilor n dimensionali şi prin $\mathcal{K}$ spaţiul cheilor. Un bloc de text clar se va nota prin M iar un bloc de text cifrat se va nota prin C.

Definiţia 1.1.1. Un cifru bloc pe n biţi este o funcţie $E : V_n \times K \to V_n$ astfel încât pentru orice cheie $k \in K$ funcţia $E(\cdot, k)$ este o funcţie inversabilă (funcţia de cifrare cu ajutorul cheii k) din V_n în V_n. Funcţia inversă este funcţia de decifrare şi va fi notată prin $D_K(\cdot) = E_K^{-1}(\cdot)$.

Reamintim că a *sparge un cifru* nu înseamnă în mod obligatoriu de a găsi o cale practică astfel încât un interceptor să recupereze textul clar numai din criptograme. În cadrul criptografiei academice, regulile sunt relaxate considerabil. A sparge un

cifru înseamnă a găsi o slăbiciune care poate fi exploatată pentru recuperare cheii şi/sau a textului cu o complexitate mai mică decât atacul brut.

1.2. Securitatea şi complexitatea atacurilor

Obiectivul unui cifru bloc este să asigure confidenţialitatea. Obiectivul corespunzător al adversarului este să descopere textul original din textul cifrat.

Un cifru bloc este *total spart* dacă a fost descoperită cheia şi spart parţial dacă adversarul poate reconstitui o parte din textul clar, dar nu şi cheia din textul cifrat.

Pentru a evalua securitatea unui cifru bloc se obişnuieşte ca întotdeauna să se presupună că adversarul:

i) are acces la toate datele transmise prin canalul de text cifrat;

ii) (presupunerea lui Kerckhoff): ştie toate detaliile despre funcţia de cifrare, mai puţin cheia secretă.

Cele mai importante clase de atacuri pentru cifrurile cu chei simetrice sunt:
- atac pe baza textului cifrat;
- atac pe baza textului clar/cifrat;
- atac pe baza textului clar/cifrat ales de criptanalist;
- atac pe baza cheilor.

1.3. Criterii de evaluare a cifrurilor bloc

Următoarele criterii pot fi folosite pentru a evalua cifrurile bloc:
- nivelul de securitate estimat (pe baza rezistenţei la anumite tipuri de atacuri);
- mărimea cheii;
- mărimea blocului de date care se cifrează;
- complexitatea funcţiei de cifrare;
- expansiunea datelor;
- propagarea erorilor;
- viteza de cifrare a datelor.

1.4. Moduri de operare

Există două moduri principale de utilizare în practică a algoritmilor simetrici: cifrarea bloc şi cifrarea secvenţială. Cifrarea bloc operează cu blocuri de text clar şi cifrat-de regulă de 64 de biţi, uneori chiar mai mari. Cifrarea secvenţială operează cu secvenţe de text clar şi cifrat de un bit sau octet. În cazul cifrării bloc, acelaşi bloc de text clar va fi cifrat de fiecare dată în acelaşi bloc de text cifrat, folosind

aceeaşi cheie. În cazul cifrării secvenţiale, secvenţele similare de text clar vor fi cifrate diferit în cazul unor cifrări repetate.

Modurile de cifrare constituie combinaţii ale celor două tipuri de bază, unele folosind metode feedback, altele realizând simple operaţii. Aceste operaţii sunt simple, deoarece securitatea este atributul cifrării şi nu al modului în care se realizează schema de cifrare. Mai mult, modul de realizare a cifrării nu duce la compromiterea securităţii date de algoritmul de bază.

Un cifru bloc cifrează textul în blocuri de n biţi de mărimi fixe. În general este folosită valoarea $n = 64$ biţi. Cele mai cunoscute moduri de operare sunt ECB (*electronic code book*), CBC (*cipher block chaining*), CFB (*cipher feedback block*) şi OFB (*output feedback block*). În funcţie de modul de operare al cifrului bloc se vor aplica atacuri specifice. Vom nota în cele ce urmează, cu E_k se notează funcţia de cifrare a blocului în timp ce cu D_k notăm funcţia de descifrare.

1.4.1. Modul ECB (electronic code-book)

Modul ECB este cea mai obişnuită formă de cifrare bloc: un bloc de text clar este transformat într-un bloc de text cifrat, fiecare bloc fiind cifrat independent şi fiecare cheie fiind diferită de celelalte. Dacă acelaşi bloc de text clar se cifrează întotdeauna în acelaşi bloc de text cifrat, teoretic este posibilă o carte de coduri în care să se facă asocierea text clar-text cifrat. Pentru blocuri de 64 de biţi rezultă un număr de 2^{64} intrări în cartea de coduri- mărime prea mare pentru a permite memorarea şi manipularea.

Modul ECB este cel mai simplu mod de lucru, fiecare bloc de text clar fiind cifrat independent. Cifrarea se poate face luând aleator blocuri din cadrul fişierului. Acest mod de cifrare este indicat pentru cifrarea documentelor care sunt accesate aleator, ca de exemplu baze de date, unde fiecare înregistrare poate fi adăugată, ştearsă, cifrată sau descifrată independent de celelalte.

Problema ridicată de ECB este aceea că, dacă un criptanalist are pentru câteva mesaje şi textul clar şi pe cel cifrat, poate afla codul, fără a deţine şi cheia de cifrare.

Padding-ul

Completarea blocurilor (*padding*) este folosită atunci când mesajele nu se împart exact în blocuri de 64 de biţi. Se completează ultimul bloc cu un model zero-unu, alternând cifrele de 0 şi 1, astfel încât blocul să devină complet.

Algoritmul ECB

Intrare: Cheia K de k biţi, mesajul clar $M = M_1, \ldots, M_t$ pe blocuri de n biţi.

Ieşire: Textul cifrat $C = C_1, \ldots, C_t$ care ulterior se descifrează pentru a descoperi textul original.

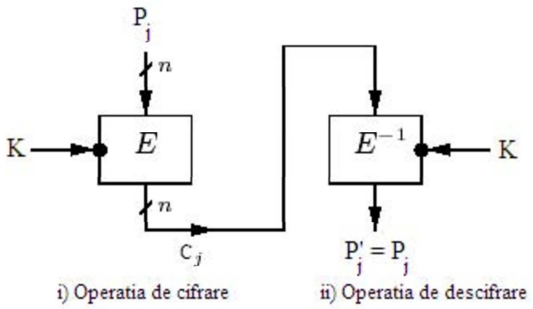

Figura 1.1: Modul de lucru ECB.

1. *Cifrarea*: pentru orice $i = 1, \ldots, t$ avem: $C_i = E_k(M_i)$.
2. *Descifrarea*: pentru orice $i = 1, \ldots, t$ avem: $M_i = D_k(C_i)$.

Proprietăţi ale modului de operare ECB

Următoarele proprietăţi rezultă din modul de construcţie al cifrului bloc tip ECB.
1. Din texte identice rezultă texte cifrate identice.
2. Dependenţa în lanţ: blocurile sunt cifrate independent de alte blocuri.
3. Propagarea erorilor: una sau mai multe erori într-un singur bloc de texte cifrate afectează descifrarea numai a acelui bloc.

1.4.2. Modul CBC (cipher-block chaining)

Acest mod foloseşte un mecanism *feedback*, deoarece rezultatul cifrării unui bloc anterior revine prin buclă şi intervine în cifrarea blocului curent. Cu alte cuvinte, blocul anterior este folosit pentru a modifica cifrarea următorului bloc. În acest fel, textul cifrat nu mai depinde doar de textul clar, ci şi de modul de cifrare al blocului anterior.

În modul CBC, textul clar, înainte de a intra în modul de cifrare propriu-zis, este însumat mod 2 cu blocul de text cifrat anterior. Figura 1.2 prezintă operaţia de cifrare/descifrare în modul CBC.

După ce blocul de text clar este cifrat, textul cifrat rezultat este stocat într-un registru al buclei de reacţie. Înainte ca următorul text clar să fie cifrat, el este sumat mod 2 cu blocul din registrul de reacţie şi devine următoarea intrare în rutina de cifrare. După cifrare , conţinutul registrului este înlocuit cu blocul cifrat. În acest fel, cifrarea blocului i depinde de toate cele $i - 1$ blocuri anterioare.

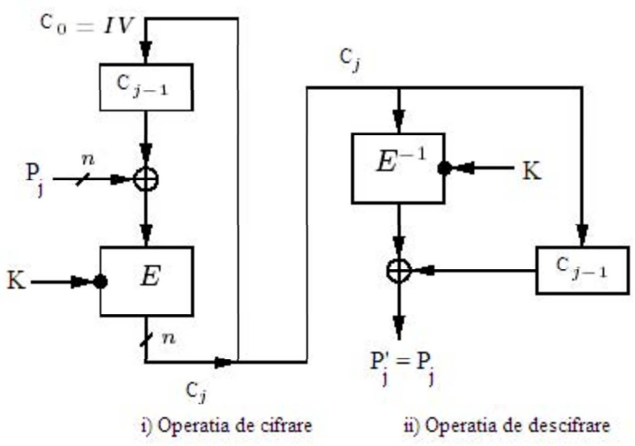

Figura 1.2: Modul de lucru CBC.

În procesul de descifrare (care este exact procesul invers cifrării), textul cifrat este descifrat normal şi depozitat în registrul de reacţie. După ce următorul bloc este descifrat, el este făcut sumă mod 2 cu conţinutul registrului de reacţie.

Din punct de vedere matematic, procesul de cifrare arată astfel:

$$C_i = E_k(P_i \oplus C_{i-1}).$$

Ecuaţiile corespunzătoare operaţiei de descifrare sunt:

$$P_i = C_{i-1} \oplus D_k(C_i).$$

Vectorul de iniţializare

Modul de lucru CBC face ca acelaşi text să se transforme în blocuri diferite de text cifrat. Două mesaje identice se vor transforma în acelaşi mesaj cifrat. Mai mult, două mesaje care încep la fel vor fi cifrate identic până la prima diferenţă. Prevenirea acestui lucru se poate face cifrând primul bloc cu un vector de date aleator. Acest bloc de date aleatoare se numeşte vector de iniţializare notat cu IV, numit şi variabilă de iniţializare sau valoare iniţială pentru înlănţuire. Vectorul IV nu are nici un înţeles de sine stătător, el doar face cifrarea oricărui mesaj, unică. Când receptorul descifrează acest bloc, el îl utilizează pentru iniţializarea registrului buclei de reacţie.

Vectorul de iniţializare nu trebuie să fie secret acesta poate fi transmis în clar împreună cu mesajul cifrat. Dacă acest lucru pare greşit, să considerăm că avem

un mesaj din câteva blocuri, $B_1, B_2, \ldots, B_i$, astfel încât B_1 este cifrat cu IV, B_2 este cifrat utilizând ca vector de iniţializare textul cifrat de la B_1,etc. Dacă avem n blocuri, sunt $n-1$ vectori de iniţializare expuşi, chiar dacă vectorul original este ţinut secret.

Padding-ul

Completarea blocurilor este analoagă celei de la modul ECB. Există însă aplicaţii în care textul cifrat trebuie să aibă aceeaşi mărime ca textul clar. Este foarte probabil ca fişierul ce urmează să fie cifrat să fie pus în aceeaşi locaţie de memorie. În acest caz rămâne un ultim bloc, mai mic, a cărui cifrare se va face diferit. Se presupune că ultimul bloc are j biţi. După cifrare, ultimul bloc întreg, se mai cifrează încă o dată, se selectează ultimii j biţi din rezultat şi se face suma mod 2 cu blocul incomplet.

Algoritmul CBC

Intrare: Cheia K pe k biţi, vectorul iniţial IV de n biţi, mesajul clar $M = M_1, \ldots, M_t$ pe blocuri de n biţi.

Ieşire: Textul cifrat $C = C_1, \ldots, C_t$ care ulterior se descifrează pentru a descoperi textul original.

1. *Cifrarea*: $C_0 = IV$, şi recursiv avem

$$C_j = E_k(C_{j-1} \oplus M_j).$$

2. *Descifrarea*: $C_0 = IV$, pentru orice $j = 1, \ldots, t$ avem

$$M_j = C_{j-1} \oplus D_k(C_j).$$

Proprietăţi ale modului de operare CBC

Au loc următoarele proprietăţi ale modului de lucru CBC.

1. Texte identice: blocuri de texte cifrate, identice, rezultă când acelaşi text este cifrat sub aceeaşi cheie şi cu acelaşi vector IV. Schimbând IV, cheia sau primul bloc de text, rezultă un text cifrat diferit.

2. Dependenţa în lanţ: mecanismul în lanţ face ca textul cifrat c_j să depindă de x_j şi de blocurile de text precedente.

3. Propagarea erorilor: o singură eroare de un bit în blocul textului cifrat c_j afectează descifrarea blocurilor c_j şi c_{j+1}.

4. Dacă o eroare (inclusiv pierderea unuia sau mai multor blocuri) apare în blocul c_j, dar nu în c_{j+1}, c_{j+2} este corect descifrat din x_{j+2}.

1.4.3. Modul CFB (cipher feedback)

În timp ce modul CBC procesează textul de n biţi odată, unele aplicaţii cer ca unităţi de r biţi ale textului să fie cifrate şi transmise fără întârziere pentru un număr fixat $r \leq n$. În acest caz se foloseşte modul CFB.

Modul CFB poate fi privit ca un mod secvenţial cu autosincronizare, însă la nivel de blocuri de text cifrat. Algoritmul CFB la nivel de bloc operează cu o coadă de mărime egală cu cea a blocului. În primă fază aceasta este iniţializată cu VI , analog modului CBC.

Dacă mărimea blocului de cifrat este n , atunci modul CFB pe n biţi are forma pentru cifrare

$$C_i = P_i \oplus E_k(C_{i-1}),$$

respectiv pentru descifrare

$$P_i = C_i \oplus E_k(C_{i-1}).$$

Figura 1.3 ilustrează această formă.

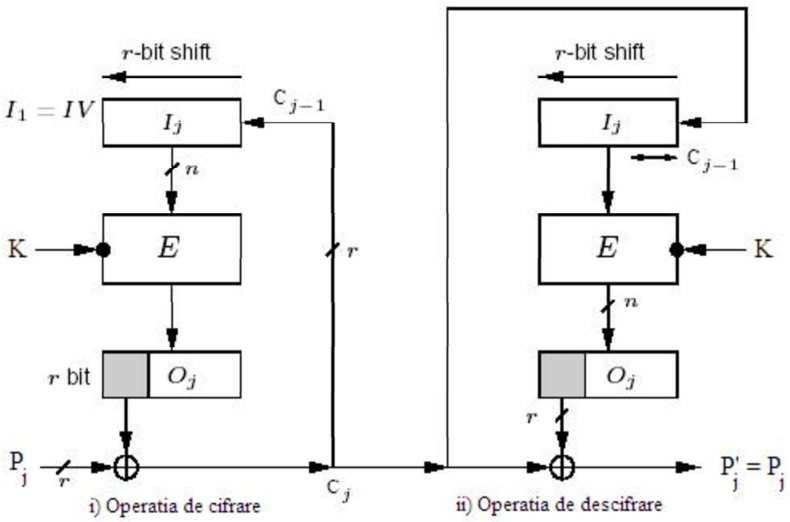

Figura 1.3: Modul de lucru CFB.

Algoritmul CFB

Intrare: Cheia K pe k biţi, vectorul iniţial IV de n biţi, textul clar pe blocuri de r biţi $M = M_1, \ldots, M_t$ cu $r \leq n$.

Ieşire: Blocuri de text cifrate pe r biţi care se descifrează pentru a descoperi textul clar.

1. *Cifrarea*: $I_1 = IV$ (I_j este valoarea de intrare a unui registru de deplasare). Fie blocul de r biţi: $x_1, \ldots, x_r$. Pentru $1 \leq j \leq r$ se execută:

a) $O_j = E_k(I_j)$ (calculează blocul cifrat de ieşire);

b) $t_j =$"cei mai la stânga" r biţi ai lui O_j;

c) $c_j = x_j \oplus t_j$;

d) $I_{j+1} = 2^r I_j + c_j \bmod 2^n$.

2. *Descifrarea*: $I_1 = IV$, pentru $1 \leq j \leq r$ calculează: $x_j = c_j \oplus t_j$, unde t_j, O_j, I_j sunt calculaţi ca mai sus.

Observaţia 1.4.1. Dacă $r = n$ atunci recurenţa corespunzătoare cifrării este:

$$C_j = M_j \oplus E_k(M_{j-1}),$$

cu

$$M_0 = IV.$$

1.4.4. Modul OFB (output feedback)

Modul de operare OFB poate fi folosit în aplicaţiile în care toate erorile de propagare trebuie să fie evitate. Este asemănător cu modul CFB şi permite cifrarea blocurilor cu diverse mărimi, dar diferă în sensul că rezultatul funcţiei de cifrare E serveşte ca feedback.Acest mod se numeşte, uneori, cu reacţie internă, deoarece bucla de reacţie este independentă atât de textul clar, cât şi de cel cifrat.

Dacă mărimea blocului este n, modul OFB pe n biţi se scrie pentru cifrare:

$$C_i = P_i \oplus S_i;$$

iar pentru descifrare:

$$P_i = C_i \oplus S_i;$$

unde

$$S_i = E_k(S_{i-1}),$$

reprezintă starea care este independentă de textul clar şi de cel cifrat.

Modul OFB pe n biţi este prezentat în figura 1.4.

Există două versiuni:

-ISO (necesită un feedback de n biţi şi este mai sigură);

-FIPS (permite $r < n$ biţi de feedback).

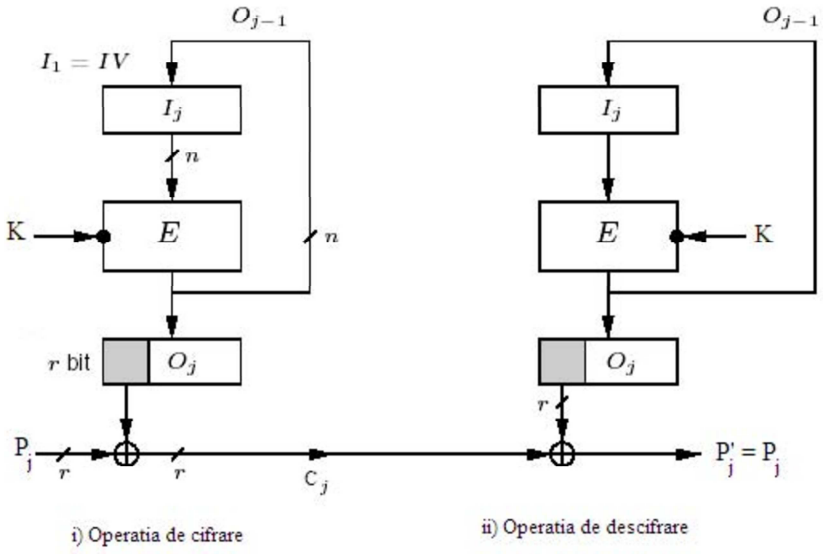

Figura 1.4: Modul de lucru OFB pe n biţi.

Algoritm OFB (pentru ISO):

Intrare: Cheia k, vectorul IV pe n biţi, blocuri de text clar pe r biţi $x_1, \ldots, x_r$ $(1 \leq r \leq n)$

Ieşire: Blocuri de text cifrate $c_1, \ldots, c_r$ pe r biţi.

1. *Cifrarea*: $I_1 = IV$. Pentru $1 \leq j \leq r$, dându-se x_j

a) $O_j = E_k(I_j)$.

b) t_j sunt cei mai de la stânga r biţi ai lui O_j.

c) $c_j = x_j \oplus t_j$.

d) $I_{j+1} = O_j$.

2. *Descifrarea*: $I_1 \leftarrow IV$. Pentru $1 \leq j \leq r$ calculăm: $x_j = c_j \oplus t_j$, unde t_j, O_j, I_j sunt calculaţi ca mai sus.

Algoritm OFB (cu feedback de r biţi) (pentru FIPS 81) :

Intrare: Cheia k pe k biţi, IV pe n biţi, blocurile de text pe r biţi $x_1, \ldots, x_r$ cu $1 \leq r \leq n$:

Ieşire: Blocurile de text cifrate $c_1, \ldots, c_r$.

Ca şi în algoritmul OFB, dar înlocuind relaţia

$$I_{j+1} = O_j$$

cu

$$I_{j+1} = (2^r I_j + t_j) \bmod 2^n.$$

Observaţia 1.4.2. O variaţie a modului de lucru OFB este *modul de lucru contor* în care trecerea de la o stare la alta este dată de funcţia de incrementare: $C_i = P_i \oplus S_i$; unde $S_i = E_k(contor[i])$, $contor[i+1] = contor[i] + 1$.

1.4.5. Modul BC (block chaining)

A folosi un algoritm bloc în modul BC înseamnă a face suma mod 2 între intrarea în blocul de cifrare şi suma mod 2 a tuturor blocurilor cifrate anterior. În acelaşi fel ca în CBC, procesul va fi declanşat de un IV.

Ecuaţiile care descriu comportarea modului de operare BC sunt, pentru cifrare

$$C_i = E_k(M_i \oplus F_i),$$

unde

$$F_{i+1} = F_i \oplus C_i,$$

cu

$$F_1 = IV.$$

Operaţia de descifrare are forma:

$$P_i = F_i \oplus D_k(C_i).$$

1.4.6. Modul BC cu sumă de control (BC-checksum)

Modul BC cu sumă de control este similar modului BC simplu, deosebirea făcându-se la ultimul bloc în care se face suma mod 2 a blocurilor clare anterioare. Modul BC cu sumă de control ne asigură de faptul că orice modificare în structura cifrată ablocurilor se va evidenţia cu ocazia descifrării ultimului bloc. Dacă acest ultim bloc are în componenţă şi elemente de integritate atunci testul de integritate se va face foarte simplu.

1.4.7. Modul OFBNLF (output feedback block with a nonlinear function)

OFBNLF este o variantă atât a modului OFB, cât şi a celui ECB, în care cheia se schimbă la fiecare bloc.

Ecuaţiile care descriu comportarea acestui mod de operare sunt, pentru cifrare:

$$C_i = E_{k_i}(P_i),$$

pentru descifrare:

$$P_i = D_{k_i}(C_i),$$

unde:

$$K_i = E_k(K_{i-1}).$$

1.4.8. Cascade de cifruri şi cifrări multiple

Există mai multe modalităţi de a combina algoritmi bloc pentru a obţine noi algoritmi de cifrare. Scopul urmărit este de a încerca să se îmbunătăţească securitatea şi prin alte mijloace decât prin scrierea unui nou algoritm. Cifrarea multiplă este una din tehnicile combinării: se utilizează un algoritm pentru a cifra acelaşi text clar de mai multe ori, cu mai multe chei.Cascadarea are acelaşi principiu, dar utilizează mai multe chei.

Definiţia 1.4.1. O cascadă de cifruri este concatenarea a $L \geq 2$ blocuri de cifruri (numite etape), fiecare cu chei independente.

Textul este intrarea primei etape iar ieşirea (rezultatul) etapei i este intrarea etapei $i+1$ şi ieşirea etapei L este ieşirea cascadei textului cifrat.

Definiţia 1.4.2. Cifrarea multiplă este similară unei cascade de L cifruri identice, dar cheile etapei nu trebuie să fie independente şi cifrurile etapei pot fi, fie un cifru bloc E, fie funcţie de descifrare corespunzătoare $D = E^{-1}$.

Cifrarea dublă

Un mod simplu de îmbunătăţire a securităţii unui algoritm bloc este de a cifra un bloc cu două chei diferite. Prima dată se cifrează blocul cu prima cheie, apoi întregul bloc de text cifrat rezultat se cifrează cu cea de a doua cheie. Descifrarea este procesul invers.

Definiţia 1.4.3. Cifrarea dublă este definită ca $E(x) = E_{K_2}(E_{K_1}(x))$.

Blocul de text cifrat rezultat din dubla cifrare ar trebui să fie mult mai greu de spart utilizându-se o căutare exhaustivă. Pentru o lungime a cheii de n biţi, faţă de 2^n variante posibile de chei la o singură cifrare , avem 2^{2^n} variante posibile, ceea ce pentru un algoritm de 64 de biţi înseamnă 2^{128} de chei posibile.

Metoda Davies-Price. Este o variantă de CBC, a cărei reprezentare matematică este următoarea:

-pentru cifrare:

$$C_i = E_{k_1}(P_i \oplus E_{k_2}(C_{i-1})),$$

-respectiv pentru descifrare:

$$P_i = D_{k_1}(C_i) \oplus D_{k_2}(C_{i-1}).$$

Double OFB/Counter. Această metodă utilizează un algoritm bloc pentru a genera două şiruri de chei ce se utilizează la cifrarea unui text clar.

Astfel putem scrie pentru variabila internă S_i :

$$S_i = E_{k_1}(S_{i-1} \oplus I_1),$$

unde (I_1 este variabilă tip contor)

$$I_1 = I_1 + 1,$$

şi respectiv pentru variabila internă T_i :

$$T_i = E_{k_2}(T_{i-1} \oplus I_2),$$

unde (I_2 este variabilă tip contor)

$$I_2 = I_2 + 1.$$

Funcţia de cifrare este:

$$C_i = P_i \oplus S_i \oplus T_i,$$

Cifrarea triplă

Definiţia 1.4.4. Cifrarea triplă este definită ca:

$$E(x) = E_{K_3}^{(3)}(E_{K_2}^{(2)}(E_{K_1}^{(1)}(x)))$$

unde $E_K^{(j)}$ este E_k sau $D_K = E_K^{-1}$.

Cifrarea triplă cu două chei. O idee mai bună, propusă de Tuchman, prelucrează un bloc de trei ori cu două chei: mai întâi cu prima cheie, apoi cu a doua, apoi din nou cu prima. El sugerează ca expeditorul să cifreze de trei ori, aşa cum specifică algoritmul, iar destinatarul, la descifrare, va urma algoritmul invers: descifrare cu prima cheie, cifrare cu a doua, descifrare cu prima.

$$C = E_{k_1}(D_{k_2}(E_{k_1}(P))),$$

$$P = D_{k_1}(E_{k_2}(D_{k_1}(C))).$$

Această metodă se mai numeşte şi EDF (*encrypt-decrypt-encrypt).*Dacă algoritmul are o cheie de n biţi, atunci schema va avea o cheie de $2n$ biţi. Modelul cifrare-descifrare a fost proiectat de IBM pentru a proteja implementările convenţionale ale algoritmului şi a fost adăugat la DES şi în standardele ISO.

Cifrarea triplă cu trei chei. În cazul cifrării triple, numărul de trei chei este cel mai potrivit. Lungimea cheii este mare, iar memorarea sa nu poate fi o problemă.

$$C = E_{k_3}(D_{k_2}(E_{k_1}(P))),$$

$$P = D_{k_1}(E_{k_2}(D_{k_3}(C))).$$

Alte variante de cifrare triplă.

Inner CBC

Se cifrează întregul mesaj în modul CBC, de trei ori diferit. Sunt necesari trei vectori de iniţializare diferiţi.

Operaţia de cifrare are următoarea formă:

$$C_i = E_{k_3}(S_i \oplus C_{i-1}),$$

unde

$$S_i = D_{k_2}(T_i \oplus S_{i-1}),$$

şi

$$T_i = E_{k_1}(P_i \oplus T_{i-1}).$$

Pentru descifrare se foloseşte următoarea formulă:

$$P_i = T_{i-1} \oplus D_{k_1}(T_i),$$

unde

$$T_i = S_{i-1} \oplus E_{k_2}(S_i),$$

şi

$$S_i = C_{i-1} \oplus D_{k_3}(C_i).$$

iar C_0, S_0 şi T_0 sunt vectori de iniţializare.

Outer CBC

Se cifrează întregul fişier în modul CBC, de trei ori diferit. Este necesar un singur vector de iniţializare.

Pentru cifrare:

$$C_i = E_{k_3}(D_{k_2}(E_{k_1}(P_i \oplus C_{i-1}))),$$

respectiv pentru descifrare:

$$P_i = C_{i-1} \oplus D_{k_1}(E_{k_2}(D_{k_3}(C_i))).$$

Ambele moduri necesită mai multe resurse decât în cazul unei singure cifrări: mai mult *hardware* sau mai mult timp.

1.5. Generarea tabelelor de substituţie

Tabele de substituţie, cunoscute sub acronimul de $S-box$, joacă un rol important în proiectarea unui cifru bloc. O tabelă de substituţie S este o funcţie bijectivă de la $GF(2^n)$ la $GF(2^n)$ (uzual $n = 8$) cu următoarele proprietăţi:

-grad mare de neliniaritate;

-rezistenţă la criptanaliza diferenţială;

-costrucţie şi computabilitate eficientă.

Următorul algoritm generează tabele de substituţie din $GF(2^n)$ cu proprietăţile menţionate mai sus. Algoritmul îşi are originea în *Nyberg* [**?**].

PAS 0. Se alege un polinom ireductibil $m(x)$ generatorul lui $GF(2^n)$, $\mathbf{A}$ matrice inversabilă din $M_{n\times n}(\mathbf{Z}_2)$, $\mathbf{b}$ vector din $\mathbf{Z}_2^n$ cu ponderea Hamming $\frac{n}{2}$.

PAS 1. Se alege $F(x)$ din $GF(2^n)$ de forma x^{-1} sau x^{2^k+1}, $k \in \mathbf{N}$.

PAS 2. Tabela S este definită prin formula:

$$S(x) = \mathbf{A}\cdot F(x) + \mathbf{b}.$$

Observaţia 1.5.1. Algoritmul AES utilizează funcţia $F(x) = x^{-1}$.

1.6. Criptanaliza diferenţială

Criptanaliza diferenţială a fost introdusă iniţial de *E. Biham* şi *A. Shamir* la începutul anilor 90. Cifrul pe care s-a aplicat, de către aceaştia, a fost standardul american de cifrare a datelor *DES* (Data Encryption Standard). În esenţă tehnica criptanalizei diferenţiale constă într-o formă particulară a tehnicii *atacului cu text clar cunoscut*: se analizează perechi de text clar cunoscut (C_i, C_j), care sunt într-o anumită relaţie (de exemplu ponderea *Hamming* a sumei mod 2 este o constantă dată sau un vector dat: $C_i \oplus C_j = D_{ij}$) şi efectul asupra textelor cifrate cu aceeiaşi cheie (de exemplu vectorul sumei mod 2 a textelor cifrate: $M_i \oplus M_j$) . Scopul urmărit este acela de a recupera (cel puţin probabilist) cheia de cifare K.

1.7. Criptanaliza liniară

Criptanaliza liniară este o tehnică prin care se urmăreşte construcţia unui sistem de ecuaţii liniare între biţii textului clar, ai textului cifrat şi ai cheii. Rezolvarea acestui sistem de ecuaţii duce la aflarea cheii de cifrare. Sistemul de ecuaţii ce se construieşte poate fi chiar un sistem probabilist, în sensul că o ecuaţie este verificată cu o anumită probabilitate.

1.8. Alte metode

În studiul cifrurilor bloc se mai pot aplica următoarele metode şi tehnici criptografice (vezi *Schneier* [**?**]):

-Metoda caracteristicilor diferenţiale condiţionate,

-Criptanaliza cu ajutorul cheilor deplasate,

-Criptanaliza diferenţial-liniară,

-Metoda combinată diferenţial-liniară,

-Metoda criptanalizei diferenţiale de ordin superior,

-Metoda criptanalizei diferenţiale de ordin superior a cifrului KN,

-Metoda aproximării multi-lineare,

-Metoda diferenţialelor trunchiate,

-Metoda generalizată a criptanalizei lineare,

-Metoda de atac prin interpolare,

-Metoda de atac prin funcţii non-surjective,

-Tehnica transformatei Walsh-Hadamard.

1.9. Implementări şi rezultate experimentale

1.9.1. Implementarea standardului de cifrare A.E.S.

Cifrul bloc AES (Advanced Encryption Standard) este noul standard de cifrare a datelor care înlocuieşte standardul DES. Standardul AES este un caz particular al algoritmului de cifrare RIJNDAEL (proiectat de Joan Daemen şi Vincent Rijmen) în sensul că are setate lungimea cheii de cifrare la 128 biţi iar dimensiunea blocului de date care se cifrează la 128, 192 sau 256 biţi. În figura 1.5 se prezintă ultimii 10 finalişti (algoritmii) care au participat la selecţia standardului. Implementarea algoritmului RIJNDAEL s-a realizat în limbajul de programare C.

Cifru	Sursă documentare
CAST-256	Entrust Technologies, Inc. (represented by Carlisle Adams)
CRYPTON	Future Systems, Inc. (represented by Chae Hoon Lim)
DEAL	Richard Outerbridge, Lars Knudsen
DFC	CNRS - Centre National pour la Recherche Scientifique - Ecole Normale Supérieure (represented by Serge Vaudenay)
E2	NTT - Nippon Telegraph and Telephone Corporation
FROG	TecApro Internacional S.A.
HPC	Rich Schroeppel
LOKI97	Lawrie Brown, Josef Pieprzyk, Jennifer Seberry
MAGENTA	Deutsche Telekom AG (represented by Dr. Klaus Huber)
MARS	IBM (represented by Nevenko Zunic)
RC6	RSA Laboratories (represnted by Matthew Robshaw)
RIJNDAEL	Joan Daemen, Vincent Rijmen
SAFER+	Cylink Corporation (represented by Dr. Lily Chen)
SERPENT	Ross Anderson, Eli Biham, Lars Knudsen
TWOFISH	Bruce Schneier, John Kelsey, Doug Whiting, David

Figura 1.5: Principalii algoritmi implicaţi în procesul de selectare a standardului A.E.S.

1.9.2. Testarea algoritmului AES

Pentru fixarea ideilor vom nota, în cele ce urmează, prin n dimensiunea blocului de date al cifrului bloc iar prin k lungimea cheii acestuia. Algoritmii candidaţi pentru standardul de cifrare bloc *AES* au fost supuşi testelor statistice (vezi *NIST 800-22* [**?**]), construcţia eşantioanelor (vezi *Soto* [**?**]) fiind realizată după cum urmează:

-*avalanşa cheii*: se fizează în mod aleatoriu un set de chei, se setează la 0−peste tot textul (text de referinţă) şi se urmăreşte efectul cifrării textului de referinţă cu ajutorul setului de chei perturbate într-un singur bit;

-*avalanşa textului de intrare*: se fizează în mod aleatoriu o mulţime de texte, se setează la 0−peste tot cheia (cheia de referinţă) şi se urmăreşte efectul produs prin cifrarea, cu cheia de referinţă, a mulţimii textelor perturbate într-un singur bit;

-*corelaţie intrare/ieşire*: se urmăreşte detectarea corelaţiilor dintre intrare şi ieşire prin utilizarea a r chei de cifrare pseudoaleatoare (de referinţă) şi a s texte de intrare pseudoaleatoare (de referinţă), cifrându-se, în modul ECB, toate cele s texte cu toate cele r chei.

-*corelaţie în modul de lucru CBC*: se fixează, în mod pseudoalatoriu un set de chei de cifrare (de referinţă), se setează la 0−peste tot vectorul de iniţializare precum şi textul clar (suficient de lung) şi se urmăreşte detecţia corelaţiei din concatenarea tuturor textelor cifrate rezultate.

-*text clar cu densitate redusă*: se testează aleatorismul statistic al cifrării tuturor textelor clare de pondere Hamming 0, 1 şi 2 cu ajutorul unui set de chei generate pseudoaleator.

-*chei cu densitate redusă*: se testează aleatorismul statistic al cifrării textelor clare generate pseudoaleator cu toate cheile de pondere Hamming 0, 1 şi 2.

-*text clar cu densitate mare*: se testează aleatorismul statistic al cifrării tuturor textelor clare de pondere Hamming n, $n-1$ şi $n-2$ cu ajutorul unui set de chei generate pseudoaleator.

-*chei cu densitate mare*: se testează aleatorismul statistic al cifrării textelor clare generate pseudoaleator cu toate cheile de pondere Hamming n, $n-1$ şi $n-2$.

1.9.3. Rezultate experimentale

Vom prezenta în cele ce urmează o serie de rezultate experimentale obţinute în cazul aplicării testelor de avalanşă strictă, imunitate la corelaţie, balans, nedegenerare precum şi a unor variante de criptanaliză diferenţială asupra algoritmului de cifrare bloc RIJNDAEL (s-au folosit pentru testare numai cheile de pondere Hamming 0 şi 1 iar ca text de referinţă textul nul).

1. Rezultatele testului de avalanşă strictă asupra algoritmului RIJNDAEL

Cheia de referinţă: cheia 0-peste tot
Textul de referinţă: 0-peste tot
Acţiune efectuată: perturbarea cu un bit a cheii de referinţă.
Dimensiunea cheii: 128 biţi
Volumul eşantionului: 128 biţi
Riscul de efectuare al testării statistice: $0,05$
Deplasarea iniţială (cuvinte ignorate): 0
Parametrul de codificare al datelor 8 biţi
Sunt afişate procentele de schimbări pe poziţiile slabe ale cheii de bază:
$60,156\%$ poziţia 31,
$60,938\%$ poziţia 37,
$60,156\%$ poziţia 51,
$40,602\%$ poziţia 125.
Numărul de poziţii slabe ale cheii 4
Decizie: algoritmul implementat indeplineşte criteriul de avalanşă strictă.
Statistica testului (procentul de respingeri) este: $-0,973329$
Indicaţii asupra atacului optimal cu chei de pondere 1: probabilităţile extreme sunt $40,625$ poziţia 125 respectiv $60,938$ poziţia 37.

2. Rezultatele testului de imunitate la corelaţie asupra algoritmului RIJNDAEL

Cheile de referinţă sunt cheile de pondere 0 şi 1
Textul de referinţă: 0-peste tot
Acţiune efectuată: perturbarea cu un bit a cheii de referinţă.
Dimensiunea cheii: 128 biţi
Deplasarea iniţială (cuvinte ignorate) 0
Parametrul de codificare al datelor 8 biţi
Volumul eşantionului: 128 biţi
Riscul de efectuare al testarii statistice 0,05
Sunt afişate procentele de schimbări de la cheile slabe:
Dezizie: cheia de pondere 0 este imună la corelaţie.
Rezultatele pentru cheile de pondere 1 sunt următoarele:
$60,156\%$ poziţia 43
$59,375\%$ poziţia 74
Numărul de corelaţii ale cheilor de pondere 1 este 2
Decizie: Algoritmul implementat indeplineşte criteriul de imunitate la corelaţie pentru cheile de pondere Hamming 1.
Statistica testului (procentul de respingeri) este: $-1,784436$
Indicaţii asupra atacului optimal cu chei de pondere 1: Probabilităţile extreme sunt $41,406\%$ poziţia 82 respectiv $60,156\%$ poziţia 43.

3. Rezultatele testului de balans asupra algoritmului RIJNDAEL
Cheile de referinţă sunt cheile de pondere 1
Textul de referinţă: 0-peste tot
Dimensiunea spaţiului de intrare este de 128 biţi
Dimensiunea spaţiului de ieşire este de 8 biţi
Depalasarea iniţială (ture in gol) 0
Parametrul de codificare al datelor: 8 biţi
Riscul de efectuare al testarii statistice $0,05$
Probabilitatea de balans este $0,5$
Sunt afisate numărul de intrări care realizează fiecare byte:
Se testează funcţia $0:\ N[0]=62,\ N[1]=66$
Statistica testului este $0,5156$
Se testează funcţia $1:\ N[0]=60,\ N[1]=68$
Statistica testului este $0,5312$
Se testează funcţia $2:\ N[0]=64,\ N[1]=64$
Statistica testului este $0,50$
Se testează funcţia $3:\ N[0]=61,\ N[1]=67$
Statistica testului este $0,5234$
Se testează funcţia $4:\ N[0]=64,\ N[1]=64$
Statistica testului este $0,5$
Se testează funcţia $5:\ N[0]=59,\ N[1]=69$
Statistica testului este $0,539$
Se testează funcţia $6:\ N[0]=62,\ N[1]=66$
Statistica testului este $0,5156$
Se testează funcţia $7:\ N[0]=54,\ N[1]=74$
Statistica testului este $0,5781$
Decizie: algoritmul implementat indeplineşte criteriul de balans.

4. Rezultatele testului de nedegenerare asupra algoritmului RIJNDAEL
Cheia de referinţă: cheia $0-$peste tot
Textul de referinţă: 0-peste tot
Dimensiunea cheii: 128 biţi
Volumul eşantionului: 128 biţi
Deplasare iniţială (cuvinte ignorate): 0
Parametrul de codificare al datelor 8 biţi
Sunt afişate punctele de degenerare ale algoritmului: nu este cazul
Numărul de puncte degenerate ale algoritmului 0
Decizie: algoritmul implementat îndeplineşte criteriul de nedegenerare.

5. Rezultatele testului de criptanaliză diferenţială asupra algoritmului RIJNDAEL

Dimensiunea cheii 128 biţi

Dimensiunea bloc de date iniţial: 128 biţi

Bloc de date iniţial 16 de 0− peste tot

Riscul de efectuare al testării statistice: $0,05$

Probabilitatea de corelare: $0,5$;

Iteraţii efectuate: 1000;

Sunt afişate cele mai semnificative cheii de bază (indice de coincidenţă extram global) pentru cheile de pondere Hamming 0 şi 1:

Cheia nulă are patternul:

Indice de coincidenţă maxim între două cifrări succesive este 0.640625 obţinut la iteraţia 914;

Indice de coincidenţă minim între două cifrări succesive este 0.343750 obţinut la iteraţia 311;

Indice de coincidenţă maxim cu textul iniţial este 0.640625 obţinut la iteraţia 890;

Indice de coincidenţă minim cu textul iniţial este 0.335938 obţinut la iteraţia 65;

Cheia cea mai slabă (corelaţie directă) este 17 cu indicele maxim de coincidenţă între două cifrări succesive 0.695313 obţinut la iteraţia 667;

Cheia cea mai slabă (corelaţie indirectă) este 13 cu indice minim de coincidenţă între două cifrări succesive 0.328125 obţinut la iteraţia 419;

Cheia cea mai slabă (autocorelaţie directă) este 88 cu indice maxim de coincidenţă cu textul iniţial 0.679688 obţinut la iteraţia 281;

Cheia cea mai slabă (autocorelaţie indirectă) este 43 cu indice minim de coincidenţă cu textul iniţial 0.320313 obţinut la iteraţia 168.

1.9.4. Interpretarea rezultatelor

După cum se observă algoritmul de cifrare RIJNDAEL îndeplineşte cerinţele formulate în cadrul capitolului 3. Acest lucru nu însemnă faptul că algoritmul este absolut sigur: *în criptografie nici o regulă nu este absolută* (Étienne Bazeries, 1901). O descriere completă a interpretării rezultatelor testelor statistico-informaţionale poate fi găsită în *Simion* [**?**].

1.10. Concluzii

Cifrurile bloc pot fi programate să funcţioneze ca un cifru flux şi reciproc (a se vedea modul de lucru OFB). Cea mai bună definiţie, care pune în evidenţă diferenţele dintre aceste două structuri de cifrare, este următoarea:

Definiţia 1.10.1. Cifrurile bloc operează asupra datelor cu transformări (fixe) asupra blocurilor de date clare; cifrurile flux operează asupra datelor cu transformări (ce variază în timp) asupra cuvintelor (biţi, octeţi, etc.) din textul clar.

În cazul aplicaţiilor practice, cifrurile bloc par mai generale (pot funcţiona în unul dintre cele patru moduri principale) iar cifrurile flux sunt mai simplu de analizat din punct de vedere matematic. O altă diferenţă este aceea că cifrurile flux cifrează sau descifrează un singur cuvânt de date la un tact, deci nu sunt optime pentru implementările software. Cifruile bloc sunt mai simplu de implementat din punct de vedere soft deoarece acestea operează cu blocuri de cuvinte de procesor deci sunt mai rapide. Pe de altă parte cifrurile flux sunt mai uşor de implementat în aplicaţiile software.

1.11. Aplicaţii

Deoarece nu există o formulă matematică universală care să poată fi aplicată în operaţia de criptanaliză, am propus ca exerciţii la acest capitol modificări ale unor algoritmi de cifruri bloc consacrate. Sunt date o serie de indicaţii precedate de o scurtă descriere a algoritmilor propiu-zişi.

Exerciţiul 1.11.1. Studiaţi următorele simplificări ale algoritmului RC5:
-RC5 cu 8 iteraţii dar fără rotaţii;
-RC5 cu 8 iteraţii iar numărul de rotaţii egal cu numărul de iteraţii.

Răspuns. În cele ce urmează facem o scurtă descriere a cifrului RC5 cu r iteraţii. Acesta are lungimea blocului de date variabilă dar vom considera în cele ce urmează că aceasta a fost setată la 64 biţi. Operaţia de cifrare foloseşte $2r+2$ chei dependente de cuvintele pe 32 biţi $S_0, S_1, S_2, \ldots, S_{2r+2}$ unde r este numărul de iteraţii. Pentru cifrare blocul de date se împarte în două părţi de 32 biţi notate cu L respectiv R (RC5 face apel la codificarea *little-endian* pentru împachetarea octeţilor în cuvinte: primul octet se transformă în cele mai puţin semnificative poziţii ale lui L, etc.). Apoi avem:

$$\begin{cases} L = L + S_0, \\ R = R + S_1. \end{cases}$$

Pentru $i = 1, \ldots, r$ se execută:

$$\begin{cases} L = ((L \oplus R) << R) + S_{2i}, \\ R = ((R \oplus L) << L) + S_{2i+1}. \end{cases}$$

Ieşirea constă în registrele L şi R. Simbolul $\oplus$ are semnificaţia sumei mod 2, simbolul $<<$ semnifică rotire circulară şi în fine simbolul $+$ are semnificaţia sumei mod 2^{32}. Operaţia de descifrare este similară (intervin operatorii $\oplus, >>$ şi $-$). Modul de construcţie al secvenţei S (care derivă din cheie) nu este esenţial în cadrul acestui exerciţiu.

Dacă setăm numărul de iteraţii $r = 8$ şi nu facem nici un fel de rotaţii atunci pentru $i = 1, \ldots, 8$ se execută:

$$\begin{cases} L = (L \oplus R) + S_{2i}, \\ R = (R \oplus L) + S_{2i+1}. \end{cases}$$

Algoritmul astfel setat nu îndeplineşte criteriul de avalanşă strictă (schimbarea unui bit în blocul de text clar produce, în medie, schimbări de 50% la ieşire). Schema de mai sus permite atacul cu ajutorul tehnicii criptanalizei liniare pentru aflarea lui S, deci a cheii efective.

Dacă setăm numărul de iteraţii $r = 8$ şi numărul de rotaţii egal cu r atunci pentru $i = 1, \ldots, 8$ se execută:

$$\begin{cases} L = ((L \oplus R) << 8) + S_{2i}, \\ R = ((R \oplus L) << 8) + S_{2i+1}. \end{cases}$$

Algoritmul astfel setat nu îndeplineşte criteriul de avalanşă strictă. Schema de mai sus permite atacul cu ajutorul tehnicii criptanalizei diferenţial/liniare pentru aflarea lui S.

Exerciţiul 1.11.2. Studiaţi următorele simplificări ale algoritmului DES:
-DES cu 12 iteraţii dar fără aplicaţiile S;
-DES cu 4 iteraţii;
-DES cu 6 iteraţii.

Răspuns. Cifrul bloc DES (proiectat în 1977) este sub controlul unei chei efective de 56 biţi (cheia de bază este de 64 biţi, 8 biţi fiind pentru detecţia erorilor) iar mărimea blocului de date este de 64 biţi. Textul clar este permutat iar apoi este împărţit în două blocuri L şi R de lungime 32 biţi. Se execută apoi iterativ operaţiile (pentru $i = 1, \ldots, \textit{numărul de iteratii}$):

$$\begin{cases} L_i = R_i, \\ R_i = L_i \oplus f(R_{i-1}, K_i). \end{cases}$$

În final textul este supus permutării inverse. Ne concentrăm asupra descrierii funcţiei $f : \mathbf{Z}_2^{32} \times \mathbf{Z}_2^{48} \to \mathbf{Z}_2^{32}$. Iniţial blocul R (32 biţi) este extins cu ajutorul

funcţiei E la un bloc pe 48 biţi care este sumat mod 2 cu cheia K (extinsă la 48 biţi cu ajutorul algoritmului de producere a subcheilor). Opt aplicaţii $S : \mathbf{Z}_2^6 \to \mathbf{Z}_2^4$ produc o ieşire pe 32 biţi care este permutată pentru a produce ieşirea finală dintr-o iteraţie. Dacă aplicaţiile S sunt fixe (se selectează 4 biţi din 6 în mod fix) atunci se poate aplica tehnica criptanalizei diferenţiale (biţii de la ieşire sunt biţii de la intrare (sumaţi mod 2 cu cheia K) dar într-o altă ordine).

Algoritmul DES cu 4 cât şi cu 6 iteraţii poate fi spart cu ajutorul tehnicii atacului cu text clar cunoscut.

Exerciţiul 1.11.3. Studiaţi regula B a algoritmului *Skipjack* cu 8 iteraţii.

Exerciţiul 1.11.4. Ce defect are un algoritm de cifrare care este închis (un algoritm de cifrare se numeşte *închis* dacă pentru orice chei k_1 şi k_2 există o cheie k_3 astfel încât pentru orice text clar M avem $E_{k_1}E_{k_2}(M) = E_{k_3}(M)$)?

Răspuns. Ca metodă de atac generică se poate opta pentru cifrarea repetitivă.

Exerciţiul 1.11.5. Aplicaţi tehnica criptanalizei diferenţiale şi criptanalizei liniare asupra algorimului FEAL.

Exerciţiul 1.11.6. Studiaţi tehnica criptanalizei diferenţiale în cazul algoritmului DES cu 16 iteraţii.

Exerciţiul 1.11.7. Aplicaţi tehnica criptanalizei liniare în cazul algoritmului DES cu 16 iteraţii.

Exerciţiul 1.11.8. Având la dispoziţie un cifru bloc $E_k(.)$ proiectaţi un cifru flux şi viceversa.

Exerciţiul 1.11.9. Scrieţi funcţia analitică a celor opt funcţii de substituţie S ale cifrului DES.

Exerciţiul 1.11.10. Fie $E_M(.)$ şi $D_K(.)$ funcţiile de cifrare respectiv descifrare ale unui cifru. Care este valoarea lui $D_K(E_K(M))$?

Notă. Descrierea algoritmilor RC5, DES, Skipjack şi FEAL poate fi găsită în *Schneier* [**?**] sau *Menenzes* [**?**].

Exerciţiul 1.11.11. Implementaţi modalităţi de testare a cifrurilor bloc.

Exerciţiul 1.11.12. Implementaţi modalităţi de generare a tabelelor de substituţie.

Exerciţiul 1.11.13. Fie $E(\cdot,\cdot)$ o funcţie de cifrare pe m biţi de cheie şi n biţi de date. Care este valoarea maximă a lui m astfel încât cheia efectivă a cifrului să fie m?